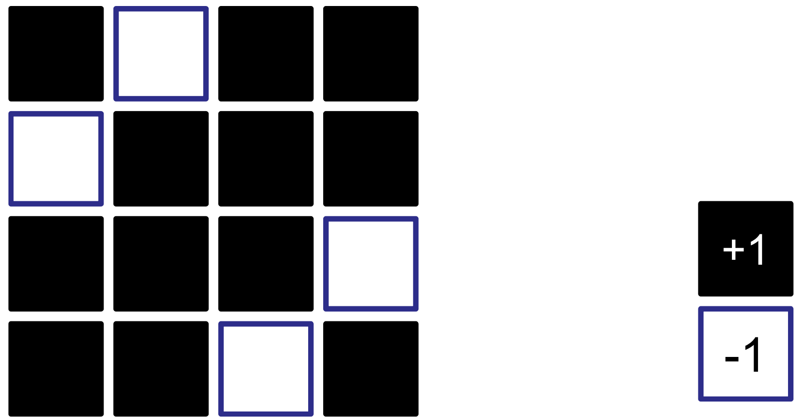
JCCI 2009

하다마드 행렬의 빠른 동치 검사

연세대학교 부호 및 암호 연구실
박기현, 송홍엽
{kh.park, hysong}@yonsei.ac.kr

하다마드 행렬 (Hadamard Matrix)

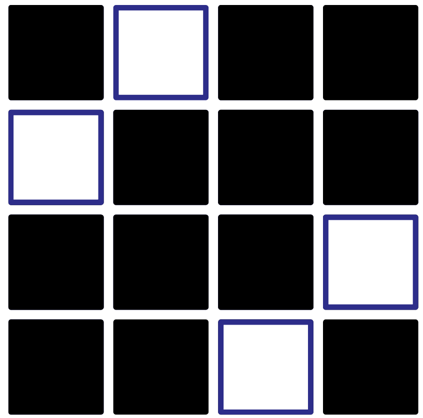


- (+1, -1) 만으로 이루어진 n x n 정방 행렬 **H**가 다음 식을 만족할 때, 행렬 **H**를 하다마드 행렬이라고 한다.

  $HH^T = nI$ **(Orthogonality of any two different row pair)**

  - I : n x n 단위행렬

하다마드-보존 연산 (Hadamard-preserving operation)



- 임의의 행 전체의 원소에 **-1**을 곱한다.
- 임의의 열 전체의 원소에 **-1**을 곱한다.
- 행의 순서를 재배치한다.
- 열의 순서를 재배치한다.
- 위 연산을 하다마드-보존 연산이라고 부르며, 위의 결과로 만들어진 하다마드 행렬을, 원래 행렬과 동치 **(Equivalent)** 라고 부른다.

하다마드 동치 행렬

- 하다마드 행렬의 각 성분의 직교성(**Orthogonality**)는 통신 신호 설계 영역에 광범위하게 사용된다.
- 동치 행렬/비동치 행렬의 조사는 행렬의 분류에 관련된 문제
  - 하다마드 행렬이 적용될 많은 시스템의 기반 이론으로 응용
- 동치 행렬에 대한 선행 연구

하다마드 행렬의 크기	비동치 행렬의 개수
1, 2, 4, 8, 12	1
16	5
20	3
24	60
28	487
>32	Unknown

하다마드 동치 검사 문제

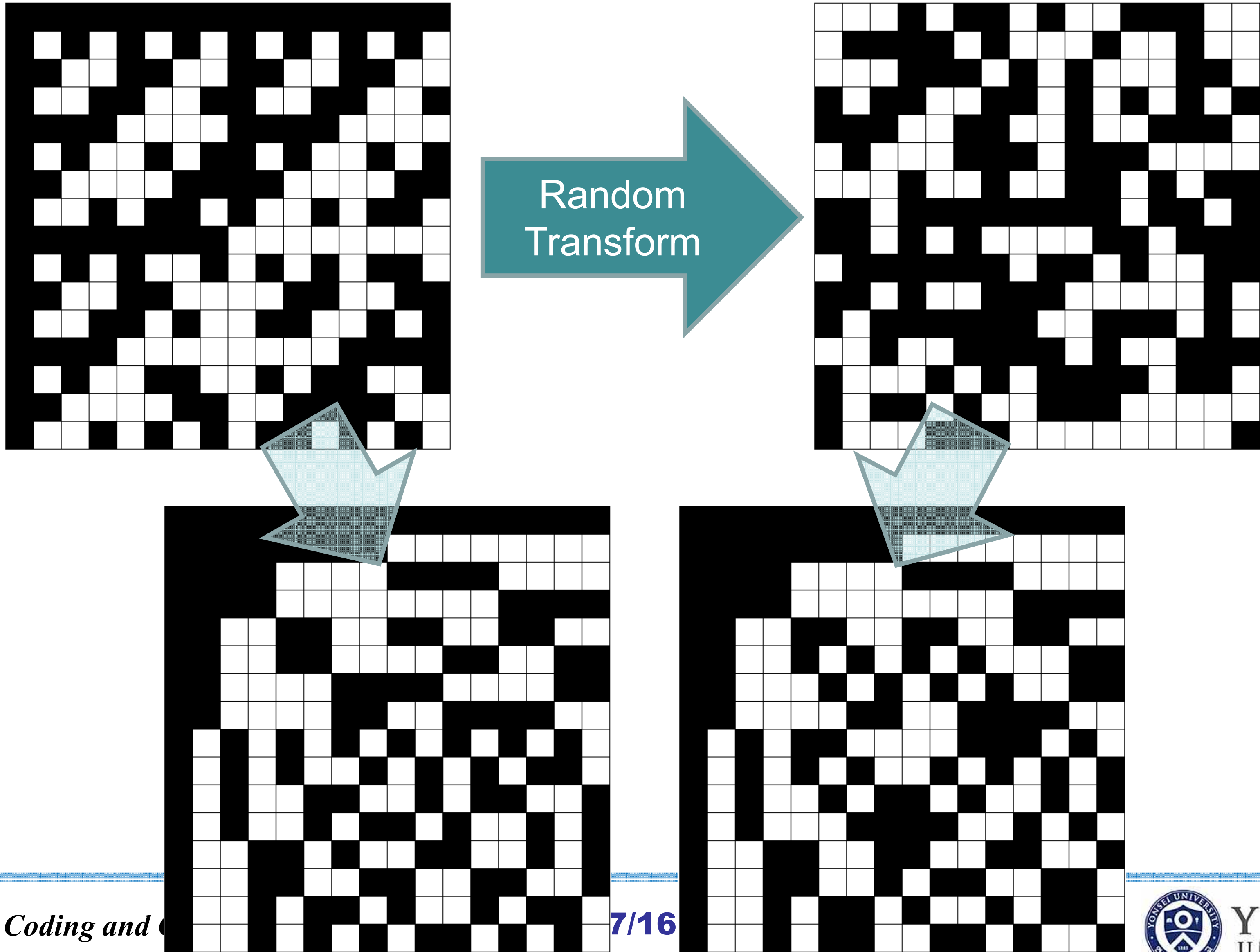
- **Easy Problem:** 주어진 하다마드 행렬에서 다른 동치의 행렬을 만드는 것
- **Hard Problem:** 주어진 두 하다마드 행렬이 동치/비동치임을 보이는 것
- 크기 n인 하다마드 행렬의 서로 다른 하다마드-동치 연산 종류:
  - Row Flip: 2^n
  - Column Flip: 2^n
  - Row Permutation: $n!$
  - Column Permutation: $n!$
  - Total: $(n!2^n)^2$
- 하다마드-보존 연산이 다르다고 결과가 항상 다른 것은 아니다.
  - Lower Bound: $n!2^n$ → 여전히 매우 큰 크기

Shin, Song의 Idea

- 하다마드 동치 행렬 집합: 서로 동치 관계에 있는 모든 하다마드 행렬의 집합
- 한 집합을 대표할 수 있는 대표값을 찾아, 그 둘을 비교하는 방식으로 동치/비동치를 판별
- 행 소트 (사전 순서로 행을 배치), 열 소트의 반복 작업을 통해서 목표 행렬을 생성
- 크기 **8,12**의 하다마드 행렬에서 유일한 대표값을 찾는 데에 성공
- 크기 **16**의 하다마드 행렬에서는 유일한 대표값을 찾는 데에 실패함

[Ref:] Min-Ho Shin and Hong-Yeop Song, "Fast algorithm of Checking Equivalence of Hadamard Matrices," 1999 한국통신학회 하계종합학술발표회, 관동대학교**, Jul. 8-10, 1999.**

Example of Failure
16x16 Varient Sylvester Hadamard



이진 행렬의 하다마드 동치

- 이진 행렬: **[+1, -1]**행렬을 **[0, 1]**행렬로 전환하여 생각
  - +1 → 0, -1 → 1

- 정의 **1.** 동일한 크기의 두 이진 행렬이 다음 두 연산의 조합으로 상호 변환이 가능할 때 이 두 행렬의 관계를 ‘하다마드-동치’라고 정의한다.
  - [OC] 하나의 행(열) 전체의 원소에 보수를 취한다.
  - [OP] 전체 행(열)의 순서를 재배치한다.

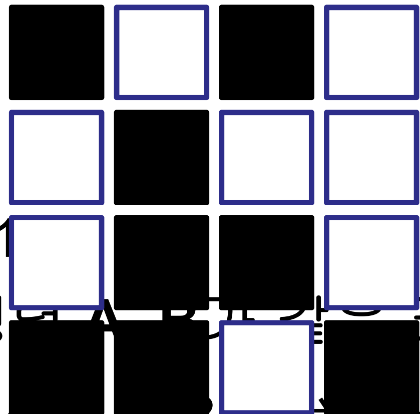
- 위 두 연산은 하다마드 보존 연산과 동치이다.

이진 행렬의 정수 변환과 순서화

- 정의 2. n x n 의 이진 행렬 $A=(a_{ij})$가 주어졌을 때, 우리는 모든 행 벡터를 연접시켜 정수로 대응할 수 있다. 이 정수 값을 ρ라고 하며, 다음과 같이 정의한다.

$$\rho(A) = \sum_{i=1}^{n}\sum_{j=1}^{n}[a_{ij}2^{n(n-i)+(n-j)}].$$

- 예:

0 1 0 1 1 0  0 1 0 0 1 0

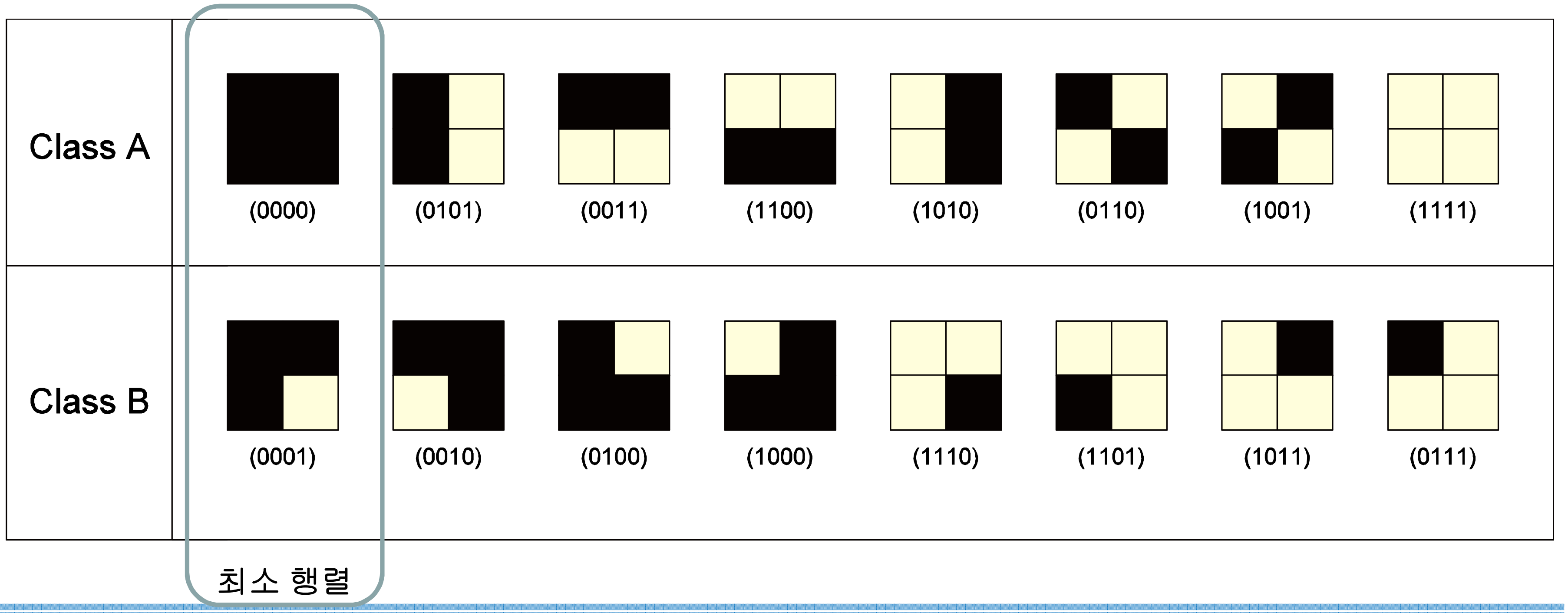
  - ρ=010110111001001

- 정리 1. 주어진 이진 행렬 A, B가 같은 크기일 때

$$\rho(A)=\rho(B) \Leftrightarrow A=B$$

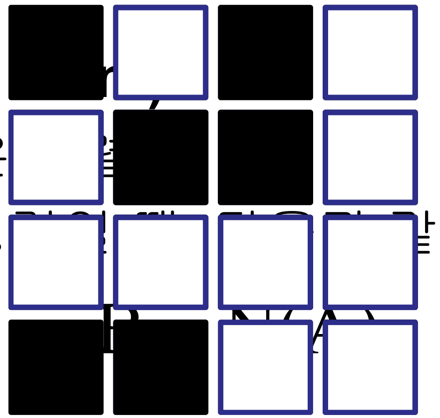
하다마드-동치 행렬 집합의 대표

- 정의 3. 모든 하다마드-동치 행렬 집합에는 최소 ρ값을 가지는 행렬이 유일하게 존재한다. 이것을 그 집합의 '최소 행렬'이라고 정의하며, 주어진 이진행렬의 '최소 행렬'이라고도 부른다.
- **Example:** 모든 **2x2** 이진 행렬. 두 개의 하다마드-동치 집합이 존재



정규화 (Normalization)

- 정규화(Normalization)
  - 주어진 이진 행렬의 첫 행과 첫 열이 모두 0이 되도록 하는 연산
  - 행렬에 OP를 적용하지 않고, 맨 앞에 1이 존재하는 행 혹은 열에 적절히 OC를 적용하는 것만으로 이 과정을 수행하는 것으로 정의
  - 예:



- 정규 행렬(Normalized
  - 정규화의 결과로 나온
  - 행렬 B가 A의 정규 행 … 같이 N을 정의한다.

$B \ldots N(A)$

조사 범위의 축소를 위한 정리

- **$(n!2^n)^2 \rightarrow (n!)^2$ 으로의 축소**
  - 정리 1: 이진 행렬의 최소 행렬은 정규 행렬이다. 따라서, 우리는 정규 행렬만을 조사하면 된다.
- **$(n!)^2 \rightarrow n!\,n$ 으로의 축소**
  - 정리 2: 주어진 행렬 A의 최소 행렬이 L일 때, $L=N(P_rAP_c)$을 만족시키는 치환 행렬(permutation matrix) P_r, P_c이 존재한다.
    - ✓ 식을 만족하는 P_r 과 P_c 의 쌍은 여러 개 존재할 수도 있다.
  - 정리 3: 위의 관계식에서 P_c 의 첫 번째 열과 P_r 전체가 결정되면, P_c 의 나머지 열과 N이 결정된다.
    - ✓ $L=N(P_rAP_c) \rightarrow N^{-1}(L)=P_rAP_c \rightarrow P_r^{-1}N^{-1}(L)=AP_c$
- **전체 조사 범위:**
  - P_r 전체: $n!$
  - P_c의 첫 번째 행의 가지수: n
  - 전체 조사 범위: 전체 연산조합 가지수 $(n!2^n)^2 \rightarrow n!\,n$

알고리즘의 복잡도

- 기본 탐색을 위한 매트릭스의 개수: $n!n$
- 매트릭스 하나 당 수행해야 할 작업
  - P_c의 나머지 열을 결정하는 작업: 소트 작업
    - ✓ 가장 빠른 소트인 퀵소트의 복잡도: $\mathcal{O}(n \log n)$
  - 연산 N을 결정하는 작업: $\mathcal{O}(n)$
  - 매트릭스 비교 작업: $\mathcal{O}(n)$
  - 총 복잡도: $\mathcal{O}(n \log n)$

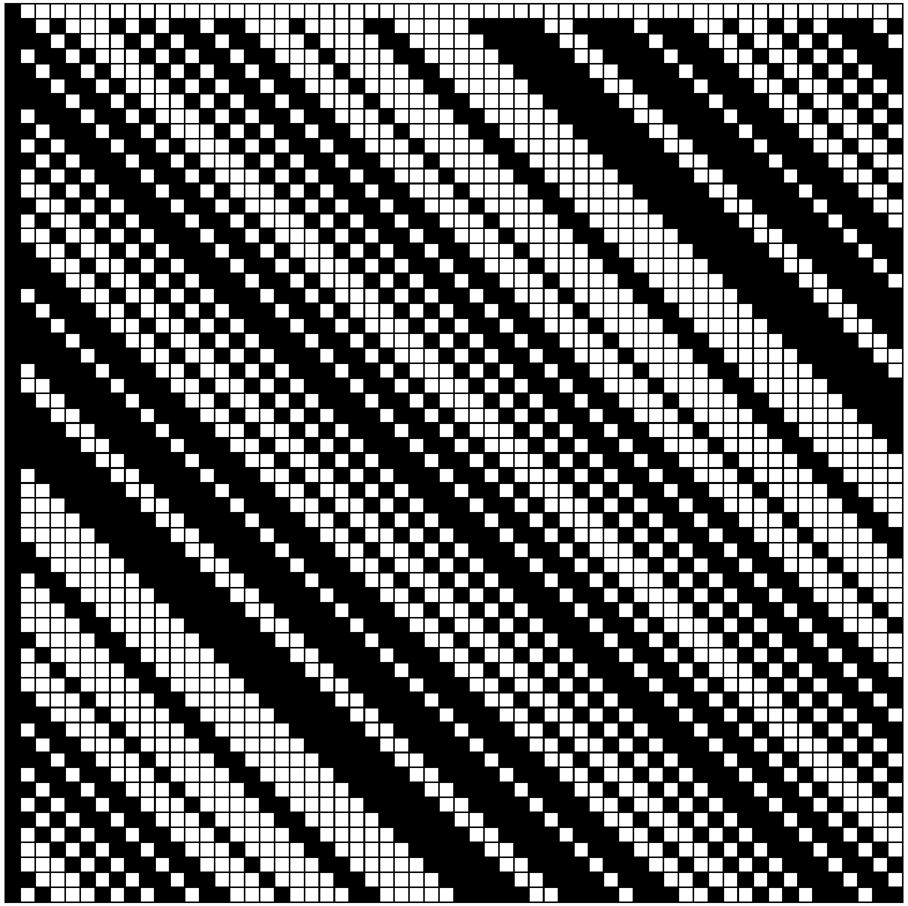
- 전체 복잡도: $\mathcal{O}(n!\, n^2 \log n)$

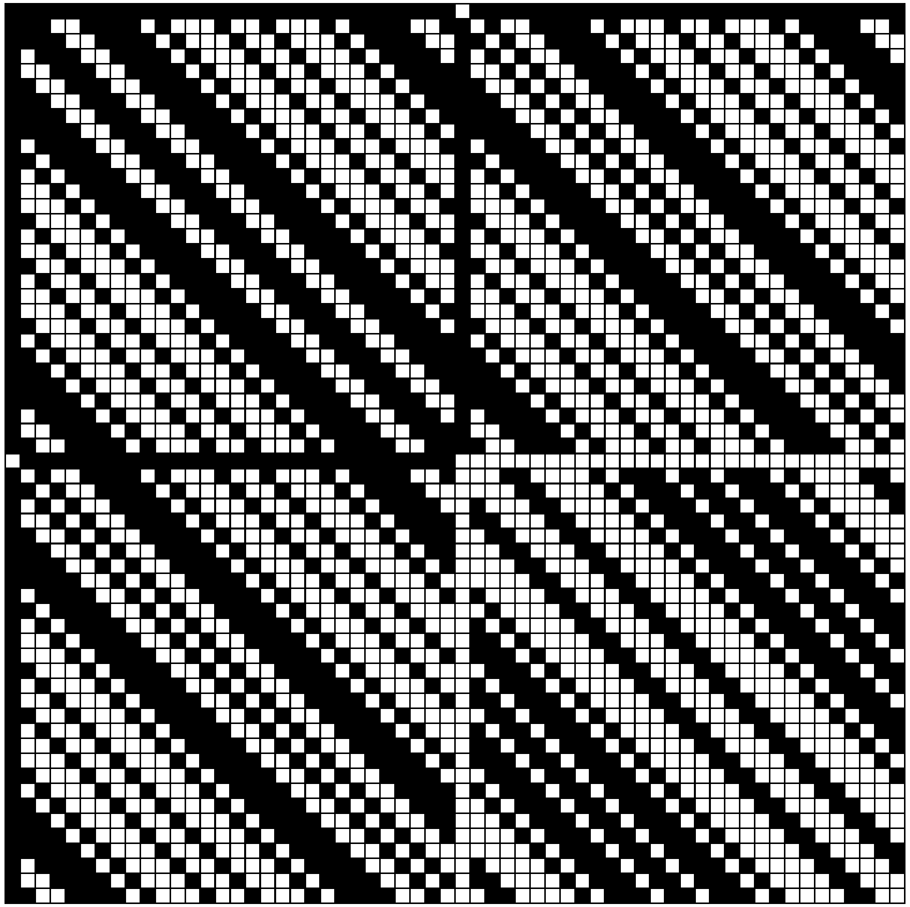
- **Back-tracking 기법의 적용: 가장 큰 복잡도인 $n!$에 적용됨**
  - Best Case: $n(n-1)$
  - Worst Case: $n!$

메인 알고리즘의 몇 가지 조사 결과와 CPU Time

Problem	Result	CPU Time
20x20 Williamson 하다마드 행렬을 임의 하다마드-보존 연산을 통해 변환한 후 알고리즘 적용	모두 동치로 판정	행렬 하나당 40초
모든 비동치 24x24 하다마드 행렬 60개의 알고리즘 적용 결과 비교	모두 비동치로 판정	행렬 하나당 2분~15분 평균 3분 가량
60x60 Paley Type I/II	비동치로 판정	Paley Type I : 30시간 Paley Type II : 3시간
32x32 Walsh (Sylvester) 와 Kronecker (M-sequence)	중간 결과 비교로 동치로 판정	약 100일 정도 소요될 것으로 추정

메인 알고리즘의 몇 가지 조사 결과의 예: 60x60 Paley Type I/II





- 하다마드 **Paley** 타입 **I(**왼쪽**)**과 **II(**오른쪽**)**의 최소 행렬**.** 두 최소 행렬의 모양이 다르다.

결론 및 고찰

- 하다마드 동치 관계에 있는 이진 행렬을 분류하는 방법을 제시
- 동치성을 확인하는 빠른 방법의 제안
- 몇몇 특별한 하다마드 행렬의 경우 소요 시간이 지나치게 길다는 단점이 있다.
  - 일반적인 환경에서 빠른 동치/비동치를 판별하기 위한 추가적인 연구 필요